TDB-ACC-NO: **NA8910133**

DISCLOSURE TITLE: Dual Digital Signal Processor Instruction TRACE Mechanism

PUBLICATION-DATA: IBM Technical Disclosure Bulletin, October 1989, US

VOLUME NUMBER: 32

ISSUE NUMBER: 5A

PAGE NUMBER: 133 - 140

PUBLICATION-DATE: October 1, 1989 (19891001)

CROSS REFERENCE: 0018-8689-32-5A-133

DISCLOSURE TEXT:

- This article describes a technique for tracing of code of two digital signal processors on a card without the involvement of external hardware devices. - Disclosed here is a dual digital signal processor instruction trace mechanism (DDSPITM) which is intended for use by digital signal processor (DSP) microcoders wishing to trace DSP code on a dual DSP based card. Access to the DDSPITM is performed by a Motorola 68000- based controller card. Access to the trace buffer and trace controls is accomplished through the 68000 reads and writes. - The DDSPITM traces only microcode branches and interrupts. This is done to optimize the 64x15 trace RAM. Tracing branches rather than actual program flow permits a larger trace to be performed within the limits of the 64 word buffer. The trace RAM is 15 bits wide, 14 bits are for storing the branch addresses, and the remaining bit is used to indicate an actual jump. ***** SEE ORIGINAL DOCUMENT ***** The following list sets forth the basic features of the DDSPITM: Trace DSP code on-card. Traces only instruction flow without direct memory access (DMA) interference. - Trace either DSP1 or DSP2. - Trace only branches. Trace before or after trigger. Support multiple triggers. Support external triggering. Stop-on-address. Single step. TRACE PROCEDURE SET UP TRACE Clear Trace RAM. Clear trace pointer (set +EN=0). Set breakpoint by setting most significant bit (MSB) of instruction located at the desired breakpoint address. Set trace control register with desired options and enable (+EN=1). Begin code execution. - Clearing the trace RAM is performed by having the 68000 write zeros in all 64 trace RAM locations. To clear the trace pointer 0 must be written into the +EN bit of the address trace

control register. Trace pointer must be cleared before a trace begins. Setting breakpoint(s) and, trace options is explained later. - OBTAINING TRACE RESULTS: Monitor end of trace (EOT) bit. Trace is complete when EOT is set: Read trace pointer. Read contents from trace RAM. Follow jump path to determine microcode flow. If is DSP halted, resume by setting 'Reset IRAM DMA Latch'. - Monitoring the EOT bit is done by having the 68000 read the 68k I/O command/status register, bit 14. To read the trace pointer, the 68000 must read the address trace control register. Bits 0 through 5 of this register will display the trace pointer. Reading the contents of trace RAM depends on what trace mode was performed. - If a trace before trigger (+TA/-TB=0) is selected, the DDSPITM is tracing continuously and wrapping around until a breakpoint is encountered. In this case, the value of the pointer is used to determine where the breakpoint occurred. Then, by reading 64 locations encountered prior to the breakpoint, a valid trace dump is produced, the only exception being that the trace RAM was not full when the breakpoint was encountered. If this happens, only the locations encountered prior to the breakpoint (i.e., those that are non-zero values) are read. If a trace after trigger (+TA/-TB=1) is selected, the DDSPITM will trace continuously until the trace RAM is full only after the breakpoint is encountered. - Once trace data is obtained, the microcode flow must be determined. The least significant bit (LSB) of the contents of each location in the trace RAM must be examined for branch detect condition to compensate for dual address condition. This condition occurs when a branch is made to a location where another branch takes place. In such a case, the address of where the first branch occurred is stored in trace RAM, followed by the address of where it branched to which is also the address of where the second branch occurred, and finally the address of where the second branch ended. Therefore, in this case these addresses were stored to represent two branches. If the two branches did not occur consecutively, the trace would normally be represented by four addresses. Shown below are how two consecutive and two non-consecutive branches would appear in a trace: Case 1 Case 2 TWO CONSECUTIVE BRANCHES TWO NON-CONSECUTIVE BRANCHES 1sb address 1sb address 1 ADDRESS1 1 ADDRESS1 1 ADDRESS2 0 ADDRESS2 0 ADDRESS3 1 ADDRESS3 0 ADDRESS4 Case 1 shows a branch occurred from ADDRESS1 to ADDRESS2 and a branch from ADDRESS2 to ADDRESS3. Case 2 shows a branch from ADDRESS1 to ADDRESS2, followed by consecutive instruction execution until the branch from ADDRESS3 to ADDRESS4. The 1sb shows where actual branches took place. - SETTING BREAKPOINTS: The trace is initially set by setting up the breakpoint (BRK) bit(s). This is accomplished by writing a '1' in the most significant bit (MSB) of the instruction whose location is the desired breakpoint. When the breakpoint is encountered in the program flow during operation, it triggers the DDSPITM. - Several triggers are possible by simply setting as many breakpoint bits as desired. The trigger will occur as soon as the first breakpoint is encountered. Fig. 1 illustrates how the **breakpoint bit** is set with the actual **instruction** in RAM. - Fig. 2 shows the DDSPITM design. Once the breakpoint(s) are

set and the desired options selected in the address trace control register, the DDSPITM is functional. During a trace, the branch detect logic senses a branch by taking the difference between the previous address and the current address. If the difference is anything other than one, a branch is detected. The branch detect logic then signals the control logic that a branch has occurred. The control logic sends the appropriate control signals to the trace RAM to store the contents of the previous address register. Since this was a branch, the branch detect bit is set next to the previous address (1sb of trace RAM indicates branch detect). After storing the previous address, the control logic signals the trace pointer to increment, thereby pointing to the next trace RAM location. - Upon the next DSP cycle, the contents of the current address register is moved to the previous address register. Once this is done, the control logic sends the appropriate signals again to the trace RAM to store the contents of the previous address register. This time the branch detect bit is set only if another branch has just occurred sequentially. After storing, the trace pointer is again incremented to point to the next RAM location. - The two DSPs are time division multiplexed on the instruction memory address bus (IMAB). The DSP-1/+2 bit in the address trace control register selects which processor to trace by selecting the appropriate clock phases. These clock phases indicate which DSP is currently on the IMAB. - ADDRESS TRACE CONTROL REGISTER: The address trace control register shown in Fig. 3 is accessible by the 68000 controller. The actual control register is located in the least significant nibble (4 bits) of this location. These four bits may be read from or written to. Bits 0 - 5 display the contents of the trace pointer during a read. LSB BIT 15: DSP-1/+2 Select which DSP to trace 0 = DSP1, 1 = DSP2 BIT 14: +TA/-TB Select trace before or after trigger. 0 = Trace before trigger. - Trace until a breakpoint is found. 1 = Trace after trigger. - Begin tracing when a breakpoint is found. Trace until trace buffer is full (64 addresses). BIT 13: +HLT on EOT Select whether or not to halt DSPs upon end of trace. 0 = continue after end of trace. 1 = halt upon end of trace. BIT 12: +EN Select to enable/disable DDSPITM. 0 = disable DDSPITM and clear trace pointer. 1 = enable DDSPITM. BITS 11 - 6: Undefined (zeroes) BITS 5 - 0: Trace Pointer (read only) TRACE EXAMPLES: The next examples all use the sample code shown below. This is a simple test loop with several branches written in DSP code. The pipeline effect of the DSP code architecture is ignored in the examples below in order to produce a clearer understanding. DSP1: SAMPLE DSP CODE 0000 BRAO LHI RO,X 0001 LHI R4,X 0002 B BRA1 0003 LH R1,O 0004 BRA1 NOP 0005 B BRA2 0006 NOP 0007 STH R1,O 0008 BRA2 B BRA3 (breakpoint set here) 0009 LHA R4,O 000A BRA3 B LOOP3 000B NOP 000C LHI R1,1 000D LHI R2,1 000E LOOP3 NOP 000F SHR R1,R 0010 NOP 0011 NOP 0012 B BRAO EXAMPLE 1: After having zeroed the contents of the trace RAM, set up the control register to trace after trigger, half-on-EOT inactive, trace DSP1, and enable trace. - The trace will begin at location BRA2 and will continue until the trace buffer is full. - TRACE RAM CONTENTS: Address Location Stored in RAM Branch

Detect Bit 0008 1 000A 1 000E 0 0012 1 0000 0 0002 1 0004 0 0005 1 --- 0008 1 ---notice consecutive branching 000A 0 EXAMPLE 2: After having zeroed the contents of the trace RAM, set up the control register to trace before trigger, half-on-EOT inactive, trace DSP1 and enable trace. - The trace will begin at the start of execution and will continue until the breakpoint has been reached. - TRACE RAM CONTENTS: Address Location Branch Detect Bit Stored in RAM 0002 1 0004 0 0005 1 0008 1 --- trace stops here because 0000 0 breakpoint encountered 0000 0

0000 0 0000 0 STOP-ON-ADDRESS AND SINGLE STEPPING: Stop-on-Address is produced by setting the control register to halt-on-EOT active, and trace before trigger. As soon as a breakpoint is encountered, the DSPs are halted. - To proceed with execution set ten=0 in the trace control register by writing '0000' to location 'XF4000'. Next set the trace control register to its previous contents with ten=1. After setting the trace control register, set the 'Reset IRAM DMA Latch' bit in the bus control command/status register. This causes the DSPs to continue executing. - For single stepping, simply set consecutive breakpoints in area where single stepping is desired. Set the trace control register to halt-on-EOT active and trace before trigger. The DSPs will halt at each breakpoint. Since the breakpoints are placed consecutively, the DSPs will, in effect, single step. To advance each time, set ten=0 in the trace control register by writing '0000' to location 'XF4000'. Next, set the trace control register to its previous contents with ten=1. After setting the trace control register, set the 'Reset IRAM DMA Latch' bit in the bus control command/status register. This causes the DSPs to continue executing and advance to the next breakpoint. - SPECIAL TRIGGERING FEATURES MULTIPLE BREAKPOINTS: Multiple triggers can be used. Simply set breakpoints at several different areas of code where tracing is desired. This feature allows the microcoder to trace code from a given subroutine or procedure. - If a breakpoint is detected in a multi-breakpoint environment with the halt-on-EOT active, the DSPs will stop execution until the 'Reset IRAM DMA Latch' bit is reset in the bus in control command/status register. Before setting the "reset IRAM DMA Latch' bit, set ten-0 in the trace control register. Next, set the trace control register to its previous contents with ten=1. After setting the trace control register, set the 'Reset IRAM DMA Latch' bit in the bus control command/status register. This causes the DSPs to continue executing and advance to the next breakpoint. Once the 'Reset IRAM DMA Latch' bit is set, the DSPs will resume execution until the next breakpoint is encountered. - EXTERNAL TRIGGERING: External triggering can be performed by attaching an external trigger source, such as a scope, logic analyzer or development system, to a VTL 'OR' gate on the card. Simply, remove ground from 'OR' gate leg and attach to trigger source. - This design disables trace during DMA. An additional bit can be added to the address trace control register to trace DMA action on the instruction memory address bus.

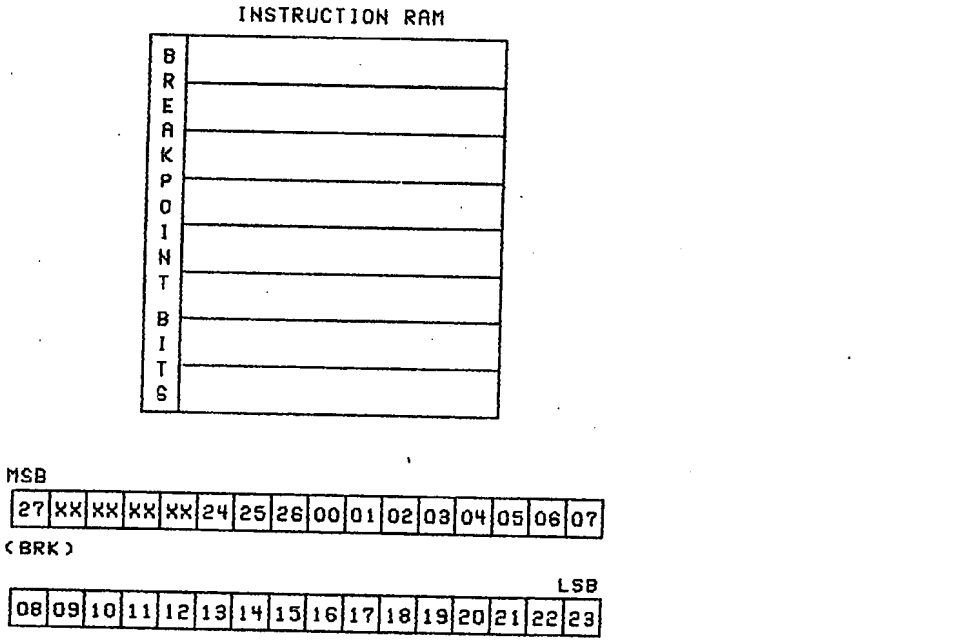

FIG. 1

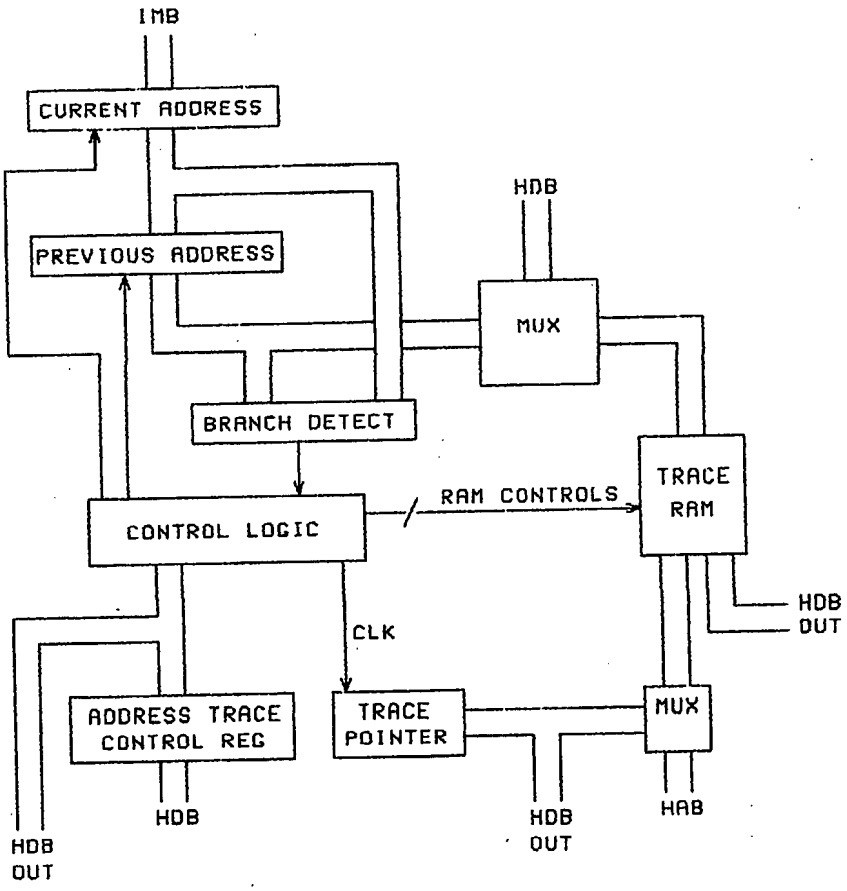

FIG. 2

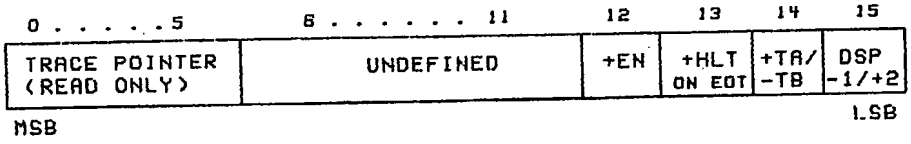

FIG. 3

6/15/07, EAST Version: 2.1.0.14